DESIGNING MULTI-CORE ARCHITECTURE USING FOLDED TORUS CONCEPT TO MINIMIZE THE NUMBER OF SWITCHES

A Thesis by

Sri RamyaChaturvedula

Bachelor of Engineering and Technology, Jawaharlal Nehru Technological University, 2009

Submitted to the Department of Electrical Engineering and Computer Science
and the faculty of the Graduate School of
Wichita State University
in partial fulfillment of
the requirements for the degree of
Master of Science

December 2011

DESIGNING MULTI-CORE ARCHITECTURE USING FOLDED TORUS CONCEPT TO MINIMIZE THE NUMBER OF SWITCHES

The following faculty members have examined the final copy of this thesis for form and content, and recommends that it be accepted in partial fulfillment of the requirement for the degree of Master of Science with a major in Computer Science.

_____
Abu Asaduzzaman, Committee Chair

_____
Ravi Pendse, Committee Member

_____
Krishna Krishnan, Committee Member

DEDICATION

To the Almighty, my loving parents, grandparents, for their encouragement throughout my studies and for incomparable advice throughout my life

ACKNOWLEDGEMENTS

I am grateful to my thesis advisor Dr. Abu Asaduzzaman for his support, encouragement, and supervision. He always has the time for guidance in spite of his busy schedule and offers me assistancewith my academics in a timely manner. I am also thankful for his great patience during my research work.

I extend my gratitude to Dr. Ravi Pendse for his valuable advice, suggestions and encouragement throughout my career at Wichita State University. It has been an honor to work for him as a graduate research assistant; I learned about professional work ethics. I also thank Dr. Krishna Krishnan for being one of my committee members and for his time and effort.

I take enormous pleasure in recognizing, with endless thanks, to all those who assisted me directly and indirectly with my experimental research. Finally, I acknowledge WSU CAPPLab research group and facilities for helping me validate and prepare my research work.

ABSTRACT

A multi-core system provides improved performance/power ratio than a single-core one. However, multi-core architecture suffers from thermal constraint and data inconsistency. Current multi-core system is not adequate to increase memory-level parallelism and cache performance due to its poor core-to-core interconnection topology. In some architecture, like MIT Raw, each node/core has computing and switching components. Switching component of such a node consumes power while the node is only computing and vice versa. In this paper, we propose a design methodology to reduce the number of switches in multi-core architecture without compromising the performance. According to this method, nodes are separated between computing cores and network switches. Using folded torus topology, we develop a scheme to connect the components (cores and switches) in the multi-core architecture. We use multi-core architectures with various numbers of nodes (cores and switches) to evaluate the proposed methodology. Using synthetic workload, we obtain the core-to-core communication delay and total power consumption for MIT RAW, Triplet Based Architecture (TriBA), Logic-Based Distributed Routing (LBDR), and the proposed architecture. Experimental results show that the proposed architecture outperforms Raw, TriBA, and LBDR by cutting down the need for the number of switches significantly. According to the results, proposed architecture reduces total power consumption approximately by 77% and average delay by 54%. Power reduction comes from the fact that number of switches is cut down. Average delay is decreased as each switch provides adequate communicate channels.

TABLE OF CONTENTS

Chapter Page

1. INTRODUCTION 1

1.1 Network Topologies....1
1.1.1 Folded Torus Topology....2
1.2Multi-Core Architecture....3
1.2.1 Switches in Multi-Core Architecture....4
1.2.2 Raw Architecture....6
1.3Problem Description....7
1.4 Contributions....8
1.5 Thesis Organization....9

2. LITERATURE SURVEY....10

2.1 Raw Architecture....10
2.2 Triplet Based Architecture....12
2.3 Logic Based Distributed Routing....14
2.4 Intra-Chip Communication....16
2.5 Multi-Core Performance/Power Ratio....16

3. PROPOSED MULTI-CORE ARCHITECTURE....18

3.1 Node Selection....18
3.1.1 Selecting Switching Nodes....19
3.1.2 Selecting Computing Nodes....20
3.1.3 Selecting Switching-Computing Nodes....20
3.2Node Connections....21
3.2.1 Connecting Switching Nodes....22
3.2.2 Connecting Computing Nodes....23
3.2.3 Connecting Switching-Computing Nodes....24
3.3 Communication Among Nodes....25

4. EVALUATION....31

4.1 Assumptions....31
4.2 Synthetic Work Load....32
4.3 Output Parameters....38
4.4 Comparison of Number of Switches with Other Architectures....38
4.5 Comparison of Power Consumption with Other Architecture....39
4.6 Comparison of Communication Delay with Other Architecture....43
4.7 Summary and Observations....47

TABLE OF CONTENTS (continued)

Chapter Page

5. CONCLUSION AND FUTURE WORK 49

5.1 Conclusion 49
5.2 Future Work 49

REFERENCES 50

LIST OF TABLES

Table Page

1a.Communication paths for Raw and TriBA in case of 16 nodes topology 33

1b.Communication paths for LBDR and proposed architectures in case of 16 nodes topology ...33

2a.Communication paths for Raw and TriBA in case of 25 nodes topology 34

2b.Communication paths for LBDR and proposed architectures in case of 25 nodes topology ...34

3a.Communication paths for Raw and TriBA in case of 36 nodes topology 35

3b.Communication paths for LBDR and proposed architectures in case of 36 nodes topology ...35

4a.Communication paths for Raw and TriBA in case of 49 nodes topology 36

4b.Communication paths for LBDR and proposed architectures in case of 49 nodes topology ...36

5a.Communication paths for Raw and TriBA in case of 64 nodes topology 37

5b. Communication paths for LBDR and proposed architectures in case of 64 nodes topology ..37

LIST OF FIGURES

Figure		Page
1.1	Ring Network Topology	2
1.2	Mesh Network Topology	2
1.3	Folded Torus Network Topology	3
2.1	Raw Architecture Tile	11
2.2	Triple Based Architecture Design	13
2.3	Memory Allocation Strategy in TriBA	14
2.4	Architecture used to Implement LBDR	15
2.5	Packet Header Format	17
3.1	Selecting Switching Nodes	19
3.2	Selecting Computing Nodes and Special Nodes	21
3.3	Connections between Switching Nodes	22
3.4	Connections between Switching Nodes and Computing Nodes	23
3.5	Connections for Special Nodes	26
3.6	Distinguishing Different Layered Nodes	27
3.7	Flowchart showing the Methodology for the Proposed Architecture	30
4.1	Numbering Convention for the Nodes	31
4.2	Graph Comparing the Number of Switching Components	39
4.3	Power Analysis for 16 Nodes	40
4.4	Power Analysis for 25 Nodes	41
4.5	Power Analysis for 36 Nodes	41
4.6	Power Analysis for 49 Nodes	42

LIST OF FIGURES (continued)

Figure Page

4.7 Power Analysis for 64 Nodes 43

4.8 Delay Analysis for 16 Nodes 44

4.9 Delay Analysis for 25 Nodes 45

4.10 Delay Analysis for 36 Nodes 45

4.11 Delay Analysis for 49 Nodes 46

4.12 Delay Analysis for 64 Nodes 46

CHAPTER 1

INTRODUCTION

1.1 Network Topologies

In everyday scenario “network” is the most general term used in the technology field. Communication among different physical nodes is defined as Network. There exist various topologies for having connections among these nodes. Multi-Core is such a technology where multiple cores communicate with each other to process a job. Here each core is considered as a node that is needed to be connected to other cores in a multi-core environment. With the advancement of Network on Chip (technology) on-chip network architecture can be explained through four parameters: topology, routing algorithm, flow control protocol and router micro architecture. Topology term in networking is defined as the how the links are connected between the nodes. Using topology of nodes all the possible paths from a particular source and destination pair can be determined. Using Routing algorithm the best path to from a source and destination pair can be identified. Using flow control protocol more details about the path selected from a source and destination pair is stored. The details include message traversal of the assigned route, when a message leaves a source node and also the time the path must be stored or buffered for future usage. Micro architecture of a networking component analyzes all the above parameters and uses it for network implementations.

In this we mainly concentrate on the topology parameter. A proper topology for a network is highly necessary for a better cost-performance on the whole network. The effect of topology while analyzing parameter is very important. Using topology of a network one can determine the number of hops a message from a source node should traverse before reaching the destination.

In general networking world different topologies like bus, mesh, ring topologies are extensively used. Figure 1.1 and Figure 1.2 shows the general ring and mesh topologies respectively.

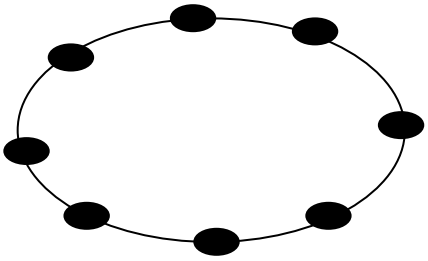

Figure 1.1:Ring Network Topology

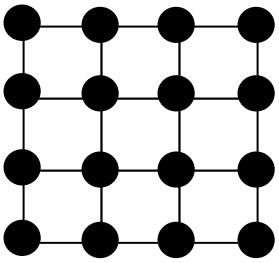

Figure 1.2:Mesh Network Topology

Each topology has its own prototype for the nodes to be connected. In multi-core architecture most of the designs have the cores connected in a mesh topology format. When these cores are connected in a multi-core environment, networking components like switches are used for communication among the cores. The topologies like Hypercube [1], Warmhole switching [2], and Crossbar switching [3] already exist in multi-core architectural designs. In this thesis we are proposing a new multi-core architectural design based on Folded Torus Topology.

1.1.1 Folded Torus Topology

Folded Torus topology is the extension of torus topology. Torus topology generally has wrap around links. When the number of nodes increases the wrap around links between the edge nodes becomes a drawback of torus topology. Hence, folded torus has a similar layout as torus, in which the links are arranged physically in a folded manner to equalize wire lengths. This can eliminate wrap around links unlike torus topology.

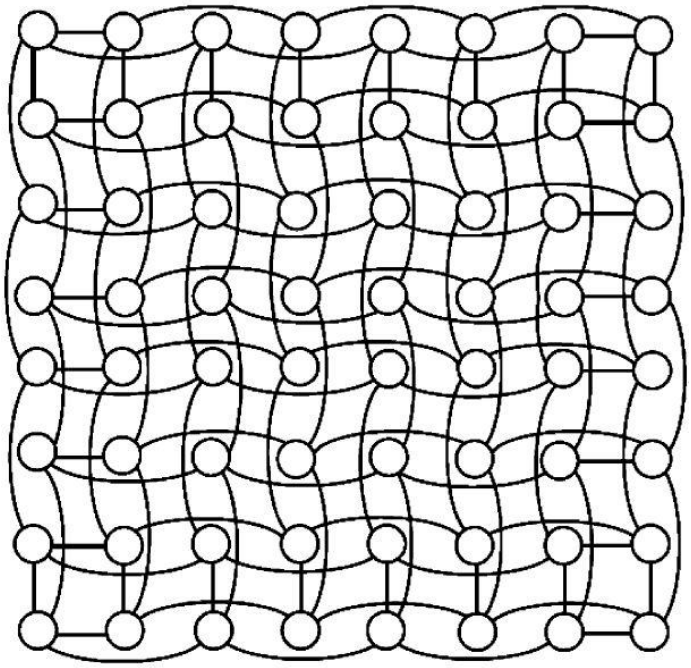

Figure 1.3:Folded Torus Network Topology [4]

As per Figure 1.3, in folded torus topology every node has a link to its every alternate node in both horizontal and vertical directions. In torus topology, source and destination pair will have lower hop count which leads to lower delay and energy.

1.2 Multi-core Architecture

To obtain more processing speed, many manufacturing companies adopt these multi-core systems in embedded systems. Multi-core systems are designed in such a way that 2 or more cores combine and work parallel to increase the speed of processing a particular job [5-12]. This kind of systemenables an embedded system for multitasking by having each core share a task or application of a job. This is not possible in single-core systems. This kind of multi tasking helps to process a job in a more efficient way. In multi-core architecture the concept of multi-core

processor improvises the computational capacity of the processors through parallel computing technology[13]. Multi-core architecture shares required resources like memory to process an application in a parallel manner. For successful and efficient processing of an application it is necessary that each core should have sufficient resources according to their respective tasks. Hence, design of multi-core architecture to utilize the available resources is very important. As discussed before nowadays multi-core designs uses bus topology to connect different cores. Designing of multi-core architecture can be done based on various components of cores in the architecture. For example designing can be done based on memory usage of architectures. In present days multi-core architectures adopt isomorphic architecture. [14-16].in this kind of architecture each core will have its own first level Cache, shared second level Cache through a bus. Also, Triplet Based Architecture [TriBA] is another kind of architecture where group of 3 cores will have common shared memory. Designs are proposed basing on the most common problem Deadlock. Deadlock is such a situation that occurs in multi-core environment where threads get stuck forever in a clash over access to shared resources like memory [17].

In [18] polling-transmission policy was discussed to solve the deadlock problem which occurs in intermediate nodes of a multi-core architecture network. This algorithm uses Hypercube topology for implementation.

Due to recent technological revolution, majority of the embedded systems are implementing more than one core for faster and efficient computations. When implementing multiple cores in a single chip it is very important to design the multi-core for efficient usage of chip volume. With this recent trends, billions of transistors are integrated on the same chip possibly. With the same capacity of chips designers are implementing multiple computing and memory cores on a single chip. This ensures computational tasks to be performed in efficient and fastest way. To design

and use such kind of systems with multiple cores requires a large design space and challenges in the research areas. This paper focuses on one of those challenges to design an efficient on-chip communication infrastructure for the multi-cores with networking components on the chip. In this era of technology multi-core processors are playing a very prominent role with their computing capabilities. Multi-core architecture has become the interesting research area to handle all the drawbacks in the present architecture like utilizing optimal space on the silicon area of a chip and minimizing heat dissipation without compromising the computing efficiency of multi-core processor. There are many designs proposed addressing the same. Some of them are like RAW architecture by MIT, Triplet Based Architecture. In a multi-core architecture for faster computational capabilities it is very much important to have efficient communication among the cores. The components that take care about this kind of communication are switches on a multi-core architecture. Generally multi-core architectures will have 2D mesh topology. When comes to the network topology there are wide range of network topologies already defined for different purposes. We are proposing a new design for multi-core architecture by concentrating on number of switching components used in the hardware of multi-core architecture. In this thesis we are using an already existing network topology called Folded Torus topology for addressing some of the issues in the existing multi-core architecture.

In [19] it is proved that the Torus based topology, with wrapper around links, will have half the network diameter. Also while accommodating the cores and switches, the bisection connections will be 2X times the number of connections on a mesh topology. Torus topology is considered to be highly efficient accepted topology for intra-chip network.

Using that topology we are proposing a design for multi-core architecture to utilize more space in a given mesh and have an efficient communication between the cores with minimal number of switching components on the mesh of cores.

1.2.1 Switches in Multi-core Architecture

It is necessary in multi-core architecture cores should pass on information to other cores on the chip to process a single application. For the cores to communicate with each other networking components like switches or routers are necessary. In this thesis we use only the term switch for networking component. Switches will actually establish a communication channel between different cores. Depending on the source and destination parameters switches will transmit packets accordingly. Network on chip (NoC) is the famous term used in nowadays multi-core environment. Very active research is going on the same NoC technology. Interconnecting different cores on the same chip for efficient communication is one of the greatest challenges of NoC. Using NoC in place of bus and ring based topologies is more flexible, scalable and reliable [20]. In [1] it is discussed that a NoC solution which switch based, happens to be the natural way for addressing the communication challenges that are due to increase in the number of cores in the multi-core environment. With increase in number of cores and usage of NoC architectures the main challenge is to improve communication efficiency among the cores. [21] Hence, the communication efficiency can be achieved through proper communication channel like switches. There are several routing algorithms proposed for NoC architectures. Wormhole switching [2], NoC Router design [22], Programmable NoC architecture [3], Hypercube-based NoC Routing algorithm [1], Logic Based Distributed Routing algorithm [23] are several proposals in the multi-core architecture for having efficient communication among the cores. In multi-core environment all the networking components are

expected to support the parallel communication patterns on demand to increase the data throughput [22]. Considering all these parameters and all the functionalities of a networking component in a multi-core environment, we are proposing a design based on number of switches that are utilized in folded torus based multi-core design. In the following sections of the document all the details about the design and other advantages of the proposed design are explained.

1.2.2 Raw Architecture

Raw Architecture from MIT is extensively analyzed for proposing this new design. This architecture uses mesh topology and processes an application. It considers the concept of tiles and each tile has a switching component, computing component and other components like cache main memory. More details about this architecture are discussed in the next chapter. The major disadvantage of this architecture in case of nxn mesh topology isthat there exists n^2 number of switches and n^2 number of computing components. Having more number of switches will increment the energy consumption. Considering this disadvantage in the following section the Problem Description is discussed.

1.3 Problem Description

In the present designs of multi-core architectures it is observed that most of the widely used topologies are mesh, ring and bus based topologies. Following are general topological views of mesh and ring topologies.

These topologies have issues like High power consumption and latency. Also, due to large number of networking components in the architecture network complexity increases. Folded

torus topology is identified as a better topology for having multiple links among the nodes in the given network. It increases the reliability of the network. As the topology of a network plays a very important role in designing multi-core architecture, the problem description of this thesis concentrates on how to propose a most reliable network topology for multi-core environment without compromising on computational efficiency. Also, we concentrated on how to minimize power consumption and latency through a better design of multi-core architecture.

In this aspect we came up with an idea of using the Folded Torus topology to increase the number of core to core connections by making some nodes as switches. This proposed design shows that we can decrease the number of switches, network power consumption and the latency. Following sections gives the detailed description of my work.

1.4 Contributions

The major contributions in my thesis are:

- Reducing the number of switches in Raw like architectures using Folded Torus topology
- Developing a methodology to compare various multi-core architecture designs to analyze performance and energy consumption. Synthetic work load is developed for different cases.
- Collaborating to develop a simulation platform to model multi-core architectures through the analysis that are calculated manually in this thesis.

All these contributions are discussed in detailed in the next chapters.

1.5 Thesis Organization

In chapter 2, we presented some of the related architectures that are already existing and well approved from various published journals and conference papers.

In chapter 3, we explained the proposed multi-core architecture and the approach to understand the methodology.

In chapter 4, we evaluated the proposed architecture by using synthetic workload and comparing with the selected existing architectures.

In chapter 5, we concluded our work by briefing the entire work and suggested future work that can be extended through this work.

CHAPTER 2

LITERATURE REVIEW

There exist various kinds on network topologies in the real world scenarios. Internetworking is also based on the network topologies. I have studied different topologies like star, mesh, bus and ring topologies. For this proposed design mesh topology is considered and customized using Folded Torus Topology. There are also other topologies like Hypercube topology that was discussed in the Introduction section. Different kinds of architectures are also studied to know the issues in the present multi-core architecture. In this chapter I would like to discuss Raw Architecture from MIT, Triplet Based Architecture (TriBA), and the architecture used for Logic Based Distributed Routing.

2.1 Raw Architecture

Raw architecture implements tile based design where each tile consists of a switching component and computing component. The main goal of RAW architecture is to improve performance over the existing architectures and provide more flexibility for the compilers of multi-core by implementing fine-grain parallelism. The Raw Architecture Workstation (Raw) is a simple, wire-efficient multi-core architecture that scales with increasing VLSI gate densities [24]. Figure 2.1 shows the components of a tile that is defined in the Raw Architecture.

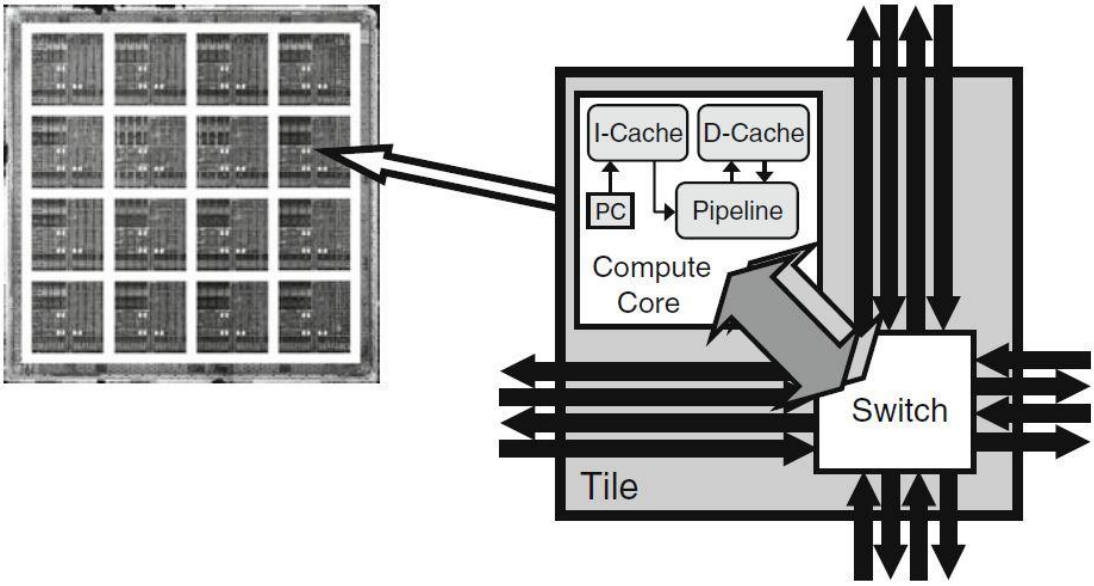

Figure 2.1: Raw Architecture Tile [25]

The tiles in this design are interconnected with several components like routers, programmable switch, switch instruction, data-memory, ALU. Firstly, Raw architecture sees to implement fine-grain parallelism in a more efficient way. Secondly, Raw architecture is designed to provide all the details about the hardware system in an architecture for the software system integrated with the architecture, such that scheduling and routing are taken care without any conflicts between the cores for shared resources. There has been some advancements in the Raw architecture [17, 25-27] proposed by Michael Bedford Taylor. This new architecture looked into concept of having static and dynamic networks for communication among the tiles. Static networks define a fixed communication channel before the compile time and the compiler exactly know where to send the message. In this static network communication each Raw tile is connected to its nearest neighbors through a series or separate, pipelines channels[17, 25-27]. In this advancement the behavior of the FPGA prototype is mainly taken care.

Also, Dynamic network communication is proposed to avoid the situations wherein the memory requirement cannot be decided before the compilation time. It uses a header and implements some protocols for dynamic routing between the tiles [26].

In spite of all the above advantages due to more switching components in a multi-core architecture the heat dissipation has become the prime concern. In our proposal we are looking into that disadvantage by decrementing the number of switching components.

2.2 Triplet Based Architecture

Triplet Based Architecture is another design model for multi-core architecture which also looked into the drawbacks of having large number of switching components. TriBA a new idea in multi-core architectures and a direct interconnection network (DIN), is compared with 2D Mesh on single chip multi core architecture. TriBA consists of a 2D grid of small, programmable processing units, each physically connected to its three neighbors so that advantageous features of group locality can be fully and efficiently utilized for getting maximum out of an on-chip Interconnection of cores. Cores on the same chip are connected via triplet-based hierarchical interconnection network (THIN), which has simple topology and computing locality characteristic [27]. TriBA basically looked at the concern where interconnected cores use the same transmission medium. To overcome the latency due to the usage of shared medium TriBA is proposed which follows hierarchical interconnected networks. In this architecture mechanisms are proposed in such a way that at each level the program decides where to send the incoming message. It decides whether to send the message to a local processor or to any other neighboring node. Efficient routing algorithms are used for this kind of mechanism and to improve the performance in the communication between the interconnected nodes in a network. Distributed Deterministic routing algorithm (DDRA) is mainly implemented for TriBA. Addressing schemes

are used for each node at each level in the hierarchical interconnection network. As TriBA is hierarchical architecture VLSI issues such as silicon area is compromised.

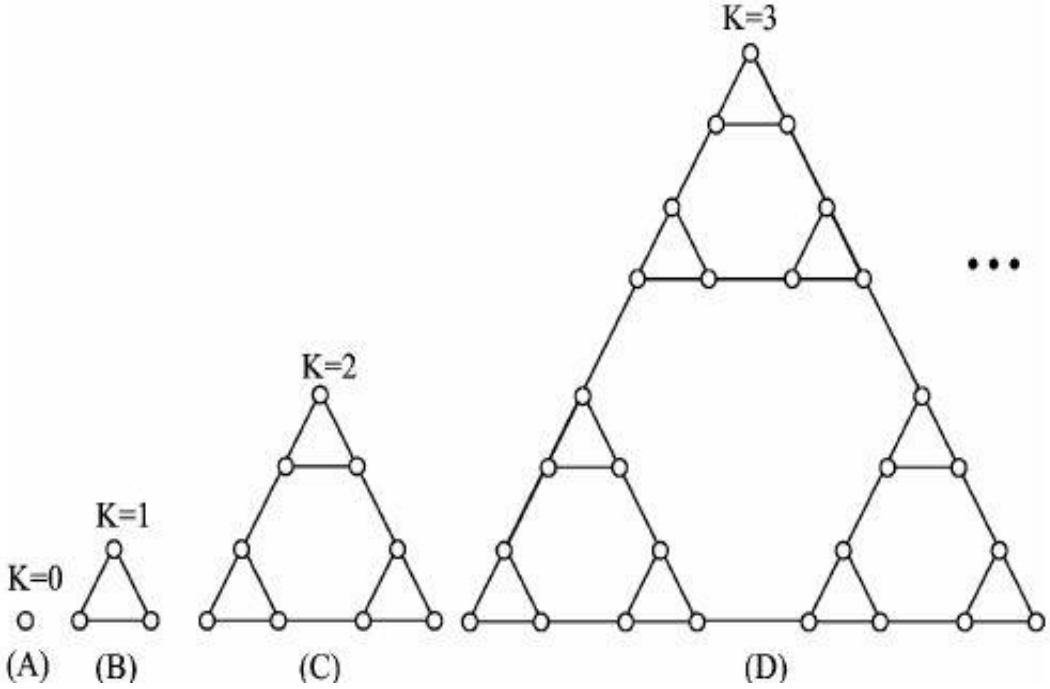

Figure 2.2:Triple Based Architecture Design [27]

As shown in the Figure 2.2, TriBA implements kind of layered architecture defining different levels. Hence, it is difficult to implement the design in 2D mesh topology. In real word scenarios 2D mesh topology is the most widely used topology is multi-core designs. Hence, we are proposing a design where maximum silicon area in a 2D mesh architecture is utilized with efficient communication among the cores.

TriBA implements a different mechanism for accessing the memory for each level of cores. There exist different memory levels depending on the type of messages each core sends.

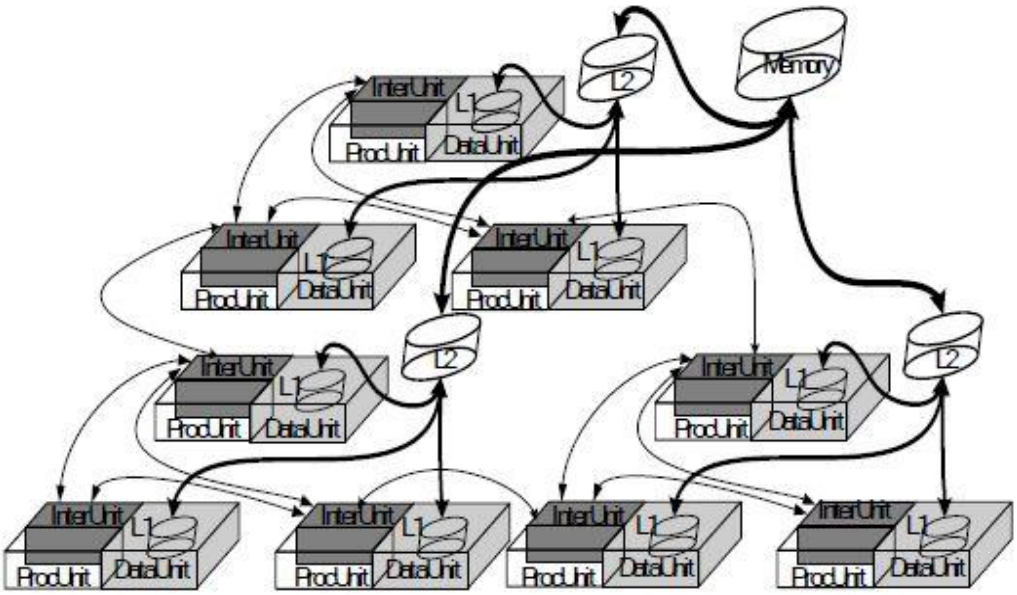

Figure 2.3:Memory Allocation Strategy in TriBA [27]

As shown in figure 2.3, three nodes (cell) are connected to each other in triangular pattern in TriBA. Each node has its local memory L1, while three nodes share a common L2 memory [25].

Prime motivation for our proposal is Raw architecture. There exist wide varieties of network topologies. In our proposal we take Folded Torus as the base topology and implement our design using the Folded Torus interconnection of nodes.

2.3 Logic-based Distributed Routing

It is known that 2D mesh topologies are generally used by designers of Network-on-Chips (NoCs). In the case of irregularities in the network it is claimed that managing routing tables as a challenging task. To overcome this complexity while dealing with the routing tables in the switches in a multi-core architecture design, a new method is proposed known as Logic-

Based Distributed Routing (LBDR). In the proposed design for LBDR, 4 cores are connected to a single switch and all switches are connected to each other.

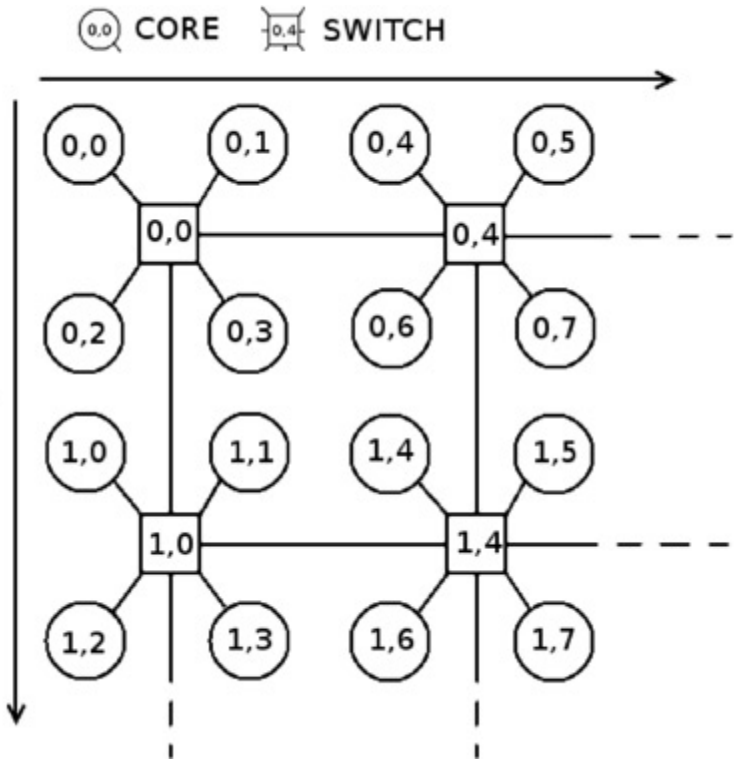

Figure 2.4:Architecture used to Implement LBDR [23]

LBDR mechanism [23] was extended to support multiple cores per switch. Figure 2.4 shows the topology where the LBDR is implemented and has multiple cores connected to each switch. In the above design it is shown that 4 cores are connected to a single switch. Motivated from the same, we proposed a new design which has multiple cores connected to the switches instead of having switching and computing components in the same node.

In the existing design every core is connected to only one switch. In the proposed design each core will be connected to minimum of 2 switches and maximum of 3 switches such that when the core finds a switch busy or dead it takes the alternative route through another switch and communicates with the other core.

2.4 Intra-chip Communication

The primary purpose of the switching components in a multi-core architecture environment is to provide intra-chip communication among the multiple cores. Intra-chip communication is defined as the communication among the computing components (cores) on a single chip for efficient throughput. In [19] the same was discussed and efficient mechanism was proposed. Folded Torus was taken as the basic architecture design with switching components at each tile for the mechanism. The main aim is to provide guaranteed throughput in terms of dead- and live-lock free and in-order data delivery, which is suitable for real-time processing applications. It was proved that power consumption with this intra-chip communication is efficient when compared to other designs.

This implement of three stage probe mechanism is to have intra-chip communication. The probe can be adaptively routed back and forth by switching nodes using a backtracked routing algorithm. It is always proven that backtracking algorithms provide efficient communication without any failure in node-node communication. As backtracking algorithms are known to have dead-lock and live-lock free communication, it is used to improve performance Intra-chip communication. In this intra-chip communication pipelined circuit-switched network is used for probe mechanism. Using circuit-switched network available path is discovered by sending three phase probe messages between each source and destination pair. For efficient communication

each probe contains a header with 3 different fields. The first field in the header contains 2 bits. There are 2 bits in the priority field, one for the interlane priority and the other for the intra-lane priority. The interlane priority bit is processed by the wrapper in order to choose an appropriate lane before sending the probe into the intra-chip network. The intra-lane priority bit is used by a switch to resolve the conflict that arises when more than one probe competes for the same output port at the switch in the circuitsetup phase.

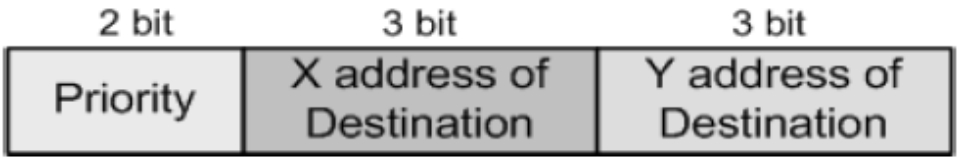

Figure 2.5:Packet Header Format [19]

As shown in the Figure 2.5 the other fields indicate Source address and destination addresses.

2.5 Multi-Core Performance/Power Ratio

Performance/power ratio is used to measure the effectiveness of a system in terms of performance (like mean delay per task) and total power consumption. Multi-core modeling and simulation techniques are presented in [28, 29] to analyze the impact of various components (like cache) on performance/power ratio. We apply some of those techniques to collect the results and evaluate our proposed torus-based multi-core architectures

CHAPTER 3

PROPOSED MULTI-CORE ARCHITECTURE

In this proposed design main goal is to reduce the number of switching components and thereby reducing the power consumption and heat dissipation. The design is mainly based on Folded Torus based network topology. In present multi-core architectures each core consists of a switching component and a computing component. Thus if nxn mesh topology is considered there exists n^2 switching components and n^2 computing components. In this proposed design, basing Torus topology a novel design is implemented for multi-core architecture to reduce the number of switches and utilize maximum silicon area on a chip. Considering nxn mesh topology every third node in a column or a row is considered as switch, such that reducing the number of switches from n^2 to considerable number of switches by following an algorithm. All the remaining number of nodes is considered as computing components or cores. In this design it is made sure that all the cores have equal number of switching components connected to have proper communication among the cores.

3.1 Node Selection

In the Proposed design unlike Raw architecture, considering the nxn mesh topology a few nodes are considered to be the switches and a few considered to be exclusively computing nodes and very few nodes are considered to be a tile like a nodes in Raw Architecture. The node selection criteria are explained in a very detailed way in the following sections.

3.1.1 Selecting Switching Components

In a given nxn topology starting from the first node, every node after 2 nodes are considered to be the switches. The same pattern is followed both in row wise and column wise. Following diagram depicts the format of selecting the switches in the case 8x8 mesh topology.

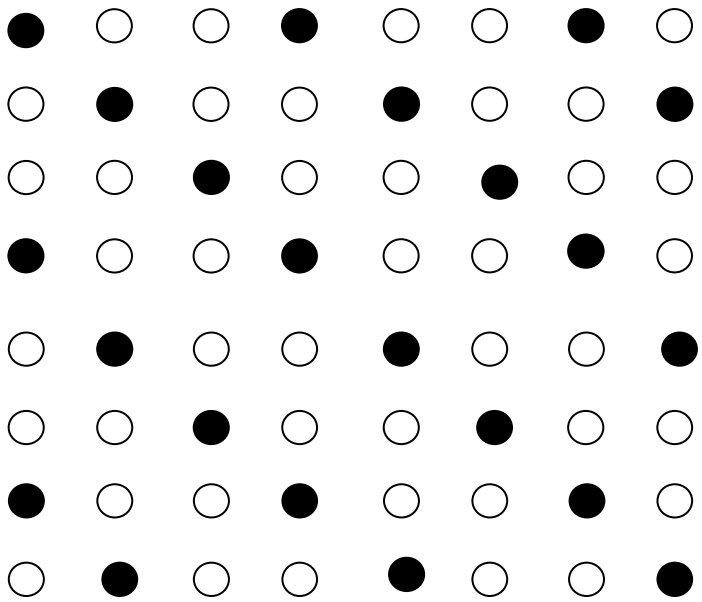

Figure 3.1:Selecting Switching Nodes

In Figure 3.1, the solid nodes indicate the switching nodes. Those nodes exclusively behave like switches. As discussed, the first node is a switch and again the fourth node in that particular column and row are switches. The same pattern is followed. Hence, the distance between 2 switches would be 3 units, considering the distance between 2 nodes is 1 unit. After all one of the main goals of this thesis is to reduce the number of switching components, we developed an algorithm to find the number of switching components in any kind of given mesh topology. In the following sections other selection criteria are discussed.

3.1.2 Selecting Computing Components

There is no big logic for selecting computing components. Except very few nodes the remaining nodes other than switches are considered to be the computing components. In the Node Selection figure the non filled nodes are computing components.

3.1.3 Selecting Switching-Computing Components

In this design the connection between each node is restricted to either in vertical or horizontal directions. There exist no connections which are diagonal. While connecting the nodes,a situation may occur where layers would form and does not connection between each layer. Hence, there comes the necessity of having a node common to those layers as both switching and computing component as in Raw Architecture. These nodes help in having full connectivity throughout the mesh. Figure 3.2 shows the connections and the special nodes that are discussed in this section.

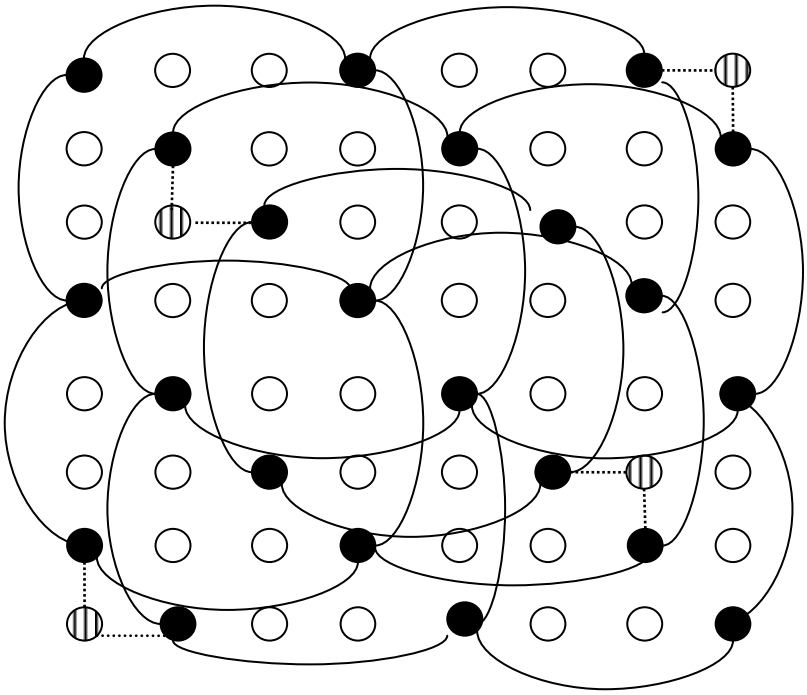

Figure 3.2: Selecting Computing Nodes and Special Nodes

In the Figure 3.2 the striped nodes are the special nodes that are used to connect 3 different layers in case of 8x8 mesh topology. The details about the connections between are discussed in the later sections.

3.2 Node Connections

The important part of this proposed design is the way the nodes are connected and the way they communicate for efficient processing of applications. The idea behind the Folded Torus Network Topology is used while defining the connections among nodes. It is important in any network topology to have proper connectivity among the switches. In multi-core architecture switches are the major components which synchronize all the computation data of the cores and provide a final result by collaborating with all the results obtained from each core involved in a process. Cores will not be able to communicate with their neighboring cores without networking

components associated with them. The proposed design ensures that every switch is connected to every other switch using different routes.

3.2.1 Connecting Switching Nodes

While defining connections between switches it is considered that every switch will have a link to its adjacent switch at a distance of 3 units (if distance between each node is 1 unit). Figure 3.3 shows how the connections exist among the switches.

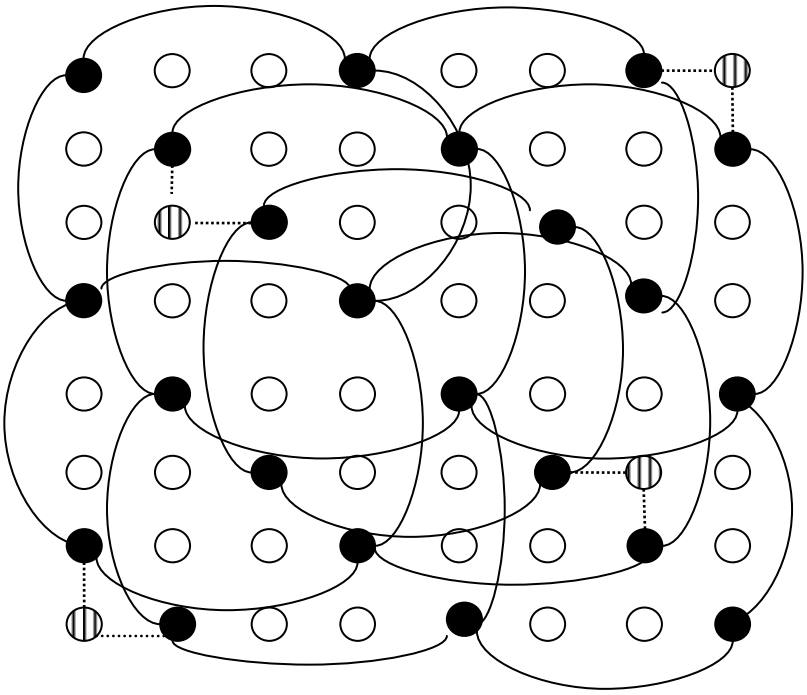

Figure 3.3: Connection between Switching Nodes

Figure 3.3 also has the special nodes just to ensure that there exists a full connectivity. All the computing nodes communicate with each only through the switches. Hence, all the switches are ensured to have full connectivity to any other switch in the network.

3.2.2 Connecting Computing Nodes

As mentioned in the previous sections, folded torus network topology idea is used to connect computing nodes and switching nodes. In folded torus topology each node will have a connection to a node at a distance of 2 units. Similarly in this proposed architecture each computing component will have a link to a switch that is at a distance of 2 units and at a distance 1 unit.

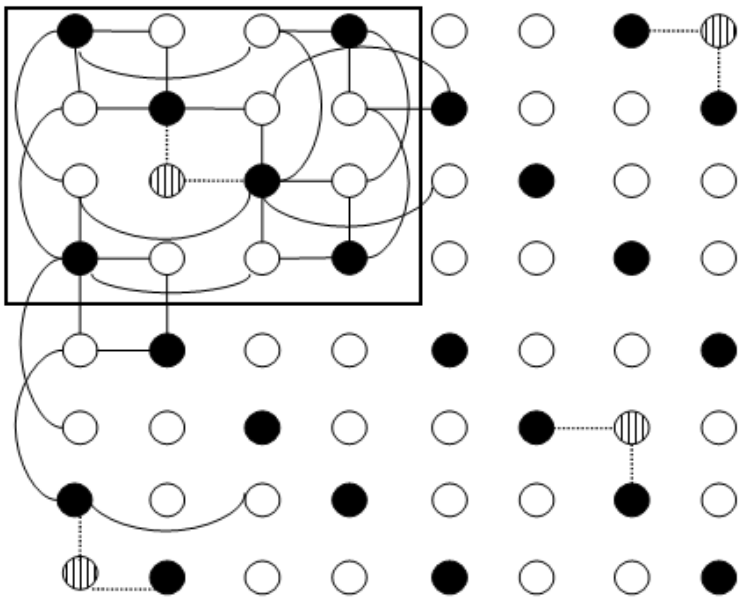

Figure 3.4: Connections between Switching Nodes and Computing Nodes

In Figure 3.4 the nodes in the square box (higlighted) is an example of nodes having connections with the switches. Computing component will have links only to the switches. They will not have any link to other computing components.

3.2.3 Connecting Switching-Computing Nodes

As discussed earlier there are very few which acts as both computing and switching nodes. These nodes will have a direct link to its adjacent node at a distance of 1 unit. In the Figure 3.4 the dotted lines between the striped nodes and normal nodes shows the connection between the special nodes and computing nodes.

All the connections among the nodes in the proposed design can be summarized using the following algorithm

Connections are made using the following algorithm:

- A connection to every switch adjacent to core
- Starting from the initial node every third node in a column or a row is considered to be the switch
- Connection to a switch at a distance of 2 physical units until the number of connections to each core reaches 3
- There exists no core-core connection
- Also every switch is connected to its nearest neighboring switches
- Every node is identified with a proper location id on the network
- We can find all switches connected to each other in a cluster form.
- Exceptions for some nodes in making them as switching components to have all the switches in the network have proper communication
- Identifying those nodes according to the location.
- Every core has equal of number switches connected to have uniform resources available to each core.

Number of switches without exceptions in any kind of nxn mesh topology can be calculated using the following loop sequence.

```
i   number of rows in a mesh topology
j:number of columns in a mesh topology
k=0; //switch counter
        for (i=1;i< number of nodes in a row;i++)
        {
                for(j=1;j<number of nodes in a column;j++)
                        {
                                Counter=i%3;
                                if (j%3==Counter)
                                {
                                sw[k] =a[i][j];
                                sw= sw+1; //counter for number of switches//
                                        k++;

                                }
                        }
                }
```

“Sw” variable gives the number of switches that can be used in a given topology.

3.3 Communication among Nodes

As mentioned earlier, it is important in any network topology to have proper connectivity among the switches. In multi-core architecture switches are the major components which synchronize all the computation data of the cores and provide a final result by collaborating with

all the results obtained from each core involved in a process. Cores will not be able to communicate with their neighboring cores without networking components associated with them. The proposed design ensures that every switch is connected to every other switch using different routes.Appropriate routing algorithms needs to be implemented for proper and efficient routing between the switches. Having different routes to all the switches help to have a deadlock free routes by using different algorithms.

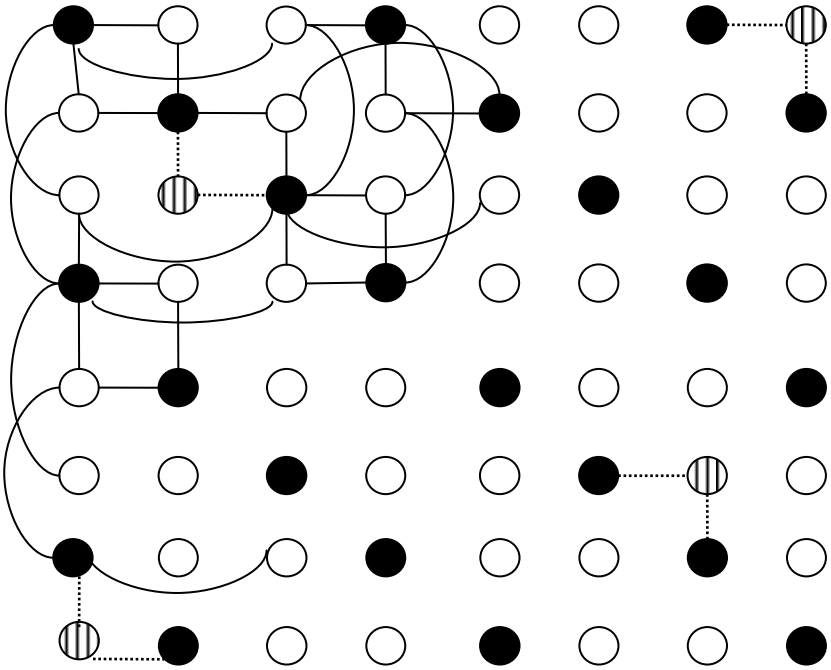

Figure 3.5: Connections for Special Nodes

Figure 3.5 depicts the part of core-switch connections. Any core will have a connection to its adjacent switch and to a switch at a distance of 2 units. As noticed before all the non filled nodes in the above topology are considered to be computing components and each core will have a

minimum of three connections to switches. This proposed design takes care that all the computing components have the uniform amount of network resources for efficient communication with the other computing components in the given architecture. All the nodes in the given nxn network are efficient used in the proposed design.

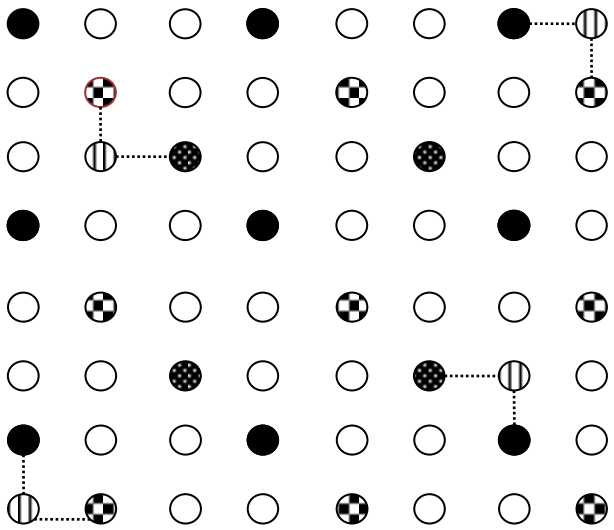

Figure 3.6: Distinguishing Different Layered Nodes

The Figure 3.6 shows all the different layers in the proposed design. All the switches which have the difference of 2 units between their positions belong to the same layer. In 8X8 mesh network there comes 3 layers. All the nodes of same layer are indicated with the same pattern in Figure 3.6. All the solid nodes indicate layer 1 nodes. As shown before striped nodes are the exception nodes to connect the different layered switches. Large checker patterned and 80 percent solid patterned nodes indicates layer 2 and layer 3 nodes respectively. The same layered nodes are

identified by the column they are present in the network. Following loop sequence is used to identify the switches of the same layer.

```
i: number of rows in a mesh topology
j:number of columns in a mesh topology
k=0; //switch counter
        for (i=0;i< number of nodes in a row;i++)
        {
                for(j=0;j<number of nodes in a column;j++)
                        {
                                Counter=i%3;
                                if (j%3==Counter)
                                {
                                sw[k] =a[i][j];
                                sw= sw+1; //counter for number of switches//
                                        k++;
                                if (Counter==0)
                               //sw[k] belongs to layer 1 with solid filling nodes
                                if(Counter==1)
                                //sw[k] belongs to layer 2 with checks filling nodes
                                if(Counter==2)
                        //sw[k] belongs to layer 3 with light dotted filling nodes color
                                }
                        }
                }
```

If a switch in a particular layer has to communicate with another switch in a different layer it has communicate through the special node that is indicated with striped pattern. Exception acts as both switching component and computing as in RAW architecture. Basically it acts as a tile in the RAW architecture. Routing algorithms can be implemented depending on the layered structure and the position of the special nodes in the mesh. The proposed design is mainly focused on 8X8 mesh topology and algorithm is generated for annxn mesh network. When compared with other existing architecture this design has the capability of utilizing more silicon area on a given chip with efficient communication among the multiple cores.

Generally it is said that one of the factors that is taken into consideration while building a network is the number of alternate routes for a given source and destination pair. In the proposed design it is assured each source and destination pair of switches has multiple routes. Hence, while defining routing algorithms one can easily find many ways to define a route either statically or dynamically. There are many algorithms proposed previously for both static and dynamic networks in multi-core architecture.

Raw Architecture Workstation also focused on same kind of discussion while proposing the design decisions. In [17, 25-27], both static network design and dynamic network design are strongly observed for the RAW architecture and new improvements are proposed for the RAW architecture.

The nodes with stripes indicate the exceptional switches and those are used to connect the switches belonging to different layers shown in Figure 3.6. Through this kind of connection the full connectivity is achieved among cores which will improve communication efficiency.

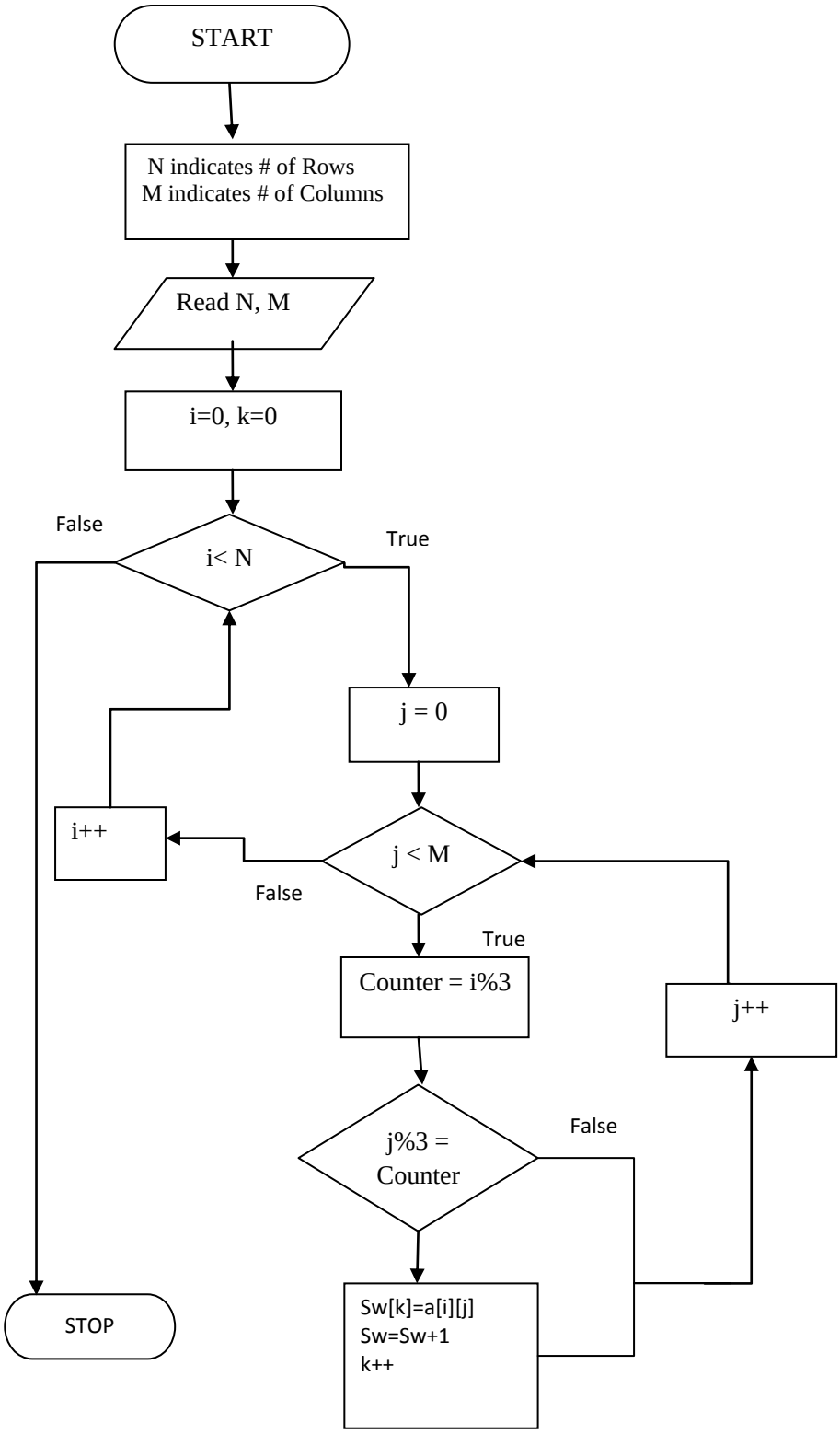

Figure 3.7: Flowchart Showing the Methodology for the Proposed Architecture

CHAPTER 4

EVALUATION

4.1 Assumptions

For each design among Raw, TriBA, design used to implement LBDR and proposed design all the nodes are numbered rowwise for Raw and proposed design. The nodes in TriBA are numbered from top to bottom in triplets. For convenience the nodes in the design that is used for LBDR are numbered in a sequential manner in multiples of 5. We considered 16 nodes, 36 nodes and 64 nodes to calculate power consumption. While calculating results for each of nxn nodes 5 cases are considered. Each case shows the number of units of power consumption when one core tries to communicate with other core in the multi-core architecture for each of the designs that are analyzed in this paper.

Following figure shows an example of the numbering convention of nodes followed for Raw architecture and proposed design for calculating results.

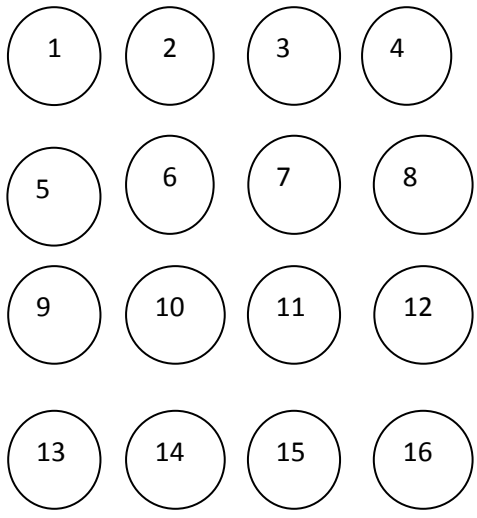

Figure 4.1: Numbering Convention for the Nodes

The Figure 4.1 shows a 4x4 mesh which contains 16 nodes. With the increase in number of nodes the nodes are renumbered as per rows and columns in the mesh. While evaluating the power consumption and communication delay are calculated for different cases when cores at random places in a given topology are communicating. It is assumed that computing components consume more power when compared to switching components in a multi-core environment. For Raw and TriBA designs it is considered that all the nodes have both switching and computing components. Hence, the power consumption by each in the path is 3 units. For switching component alone power consumption is considered as 1 unit. As discussed power consumption for computing component alone is considered as 2 units (>1). Thus, having all assumptions in place and considering different cases evaluation of new design is explained in the following sections.

4.2 Synthetic Work Load

In this section, all the different cases for each kind of topology of Raw, TriBA, LBDR and proposed are tabulated. These tables are used for evaluation of power consumption and communication delay for the 4 architectures.

Following table indicates has 6 columns and 5 rows. The first column indicates the case numbers. The second column indicates the source and destination nodes for that particular case. The other columns indicate the node numbers in each of the architectures, which a packet has to traverse from a source to destination. As assumed for Raw and TriBA architectures all the nodes are considered to have switching component and computing component. But in case of LBDR and Proposed the symbol “sw” differentiates the core node and computing node. The node with “sw” beside it is considered to be the switching node. And accordingly all assumptions are considered while evaluating.

Table 1a: Communication paths for Raw and TriBA in case of 16 nodes topology

Communication paths for 16 Nodes			
	Source-Destination	Raw	TriBA
Case 1	Node 2 – Node 15	2,3,7,11,15	2,4, 6, 13 ,15
Case 2	Node 3- Node 14	3,2,6,10,14	3,2,4,6,13,14
Case 3	Node 7 – Node 15	7,11,15	7,8,6,13,15
Case 4	Node 2 – Node 10	2,6,10	2,4,5,10
Case 5	Node 8 – Node 14	8,12,16,15,14	8,6,13,14

Table 1b: Communication paths for LBDR and Proposed Architectures in case of 16 nodes topology

Communication paths for 16 Nodes			
	Source-Destination	LBDR	Proposed
Case 1	Node 2 – Node 15	2,1(sw),6(sw),11(sw),15	2,1(sw),13(sw) ,15
Case 2	Node 3- Node 14	3,1(sw),6(sw),11(sw),14	3,1(sw),13(sw),14
Case 3	Node 7 – Node 15	7, 6(sw),11(sw),15	7,11(sw),15
Case 4	Node 2 – Node 10	2,1(sw),6(sw),10(sw)	2,6(sw),10
Case 5	Node 8 – Node 14	8,6(sw),11(sw),14	8,4(sw),16(sw),14

Tables 1a and 1b show the node-node communication in case of 16 nodes topologies for all the 4 architectures

Table 2a:Communication paths for Raw and TriBA in case of 25 nodes topology

Communication paths for 25 Nodes			
	Source-Destination	Raw	TriBA
Case 1	Node 2 – Node 24	2,7,12,17, 22,23, 24	2,4,5,10,11, 22,24
Case 2	Node 3- Node 23	3,8,13,18,23	3,2,4,5,10,11,22, 23
Case 3	Node 8 – Node 24	8,13,18,23, 24	8,6,5,10,11, 22,24
Case 4	Node 2 – Node 20	2,7,12,17,18,19,20	2,3,7,19,20
Case 5	Node 9 – Node 23	9,14,19, 24,23	9,8,6,5,10,11, 22,23

Table 2b: Communication paths for LBDR and Proposed Architectures in case of 25 nodes topology

Communication paths for 25 Nodes			
	Source-Destination	LBDR	Proposed
Case 1	Node 2 – Node 24	2,1(sw),21(sw),24	2,7(sw),22(sw) ,24
Case 2	Node 3- Node 23	3,1(sw),21(sw),23	3,13(sw),23
Case 3	Node 8 – Node 24	8,6(sw),1(sw),21(sw),24	8,7(sw),22(sw),24
Case 4	Node 2 – Node 20	2,1(sw),6(sw),11(sw),16(sw),20	2,7(sw),10(sw),20
Case 5	Node 9 – Node 23	9,6(sw),1(sw), 21(sw), 23	9,10(sw), 5(sw),23

Tables 2a and 2b show the node-node communication in case of 25 nodes topologies for all the 4 architectures.

Table 3a: Communication paths for Raw and TriBA in case of 36 nodes topology

Communication paths for 36 Nodes			
	Source-Destination	Raw	TriBA
Case 1	Node 2 – Node 35	2,8,14,20,26,32,33,34,35	2,4,6,13,15,31,33,35
Case 2	Node 3- Node 34	3,9,15,21,27,33,34	3,7,9,19,20,34
Case 3	Node 9 – Node 35	9,15,21,27,33, 34,35	9,19,20,24,35
Case 4	Node 2 – Node 30	2,8,14,20,26,27,28,29,30	2,4,6,13,14,28,30
Case 5	Node 12 – Node 34	12,18,24,30,36, 34	12,14,15,17,18,20, 34

Table 3b: Communication paths for LBDR and proposed architectures in case of 36 nodes topology

Communication paths for 36 Nodes			
	Source-Destination	LBDR	Proposed
Case 1	Node 2 – Node 35	2,1(sw),21(sw),26(sw),31(sw),35	2,8(sw),11(sw),29(sw), 35
Case 2	Node 3- Node 34	3,1(sw),21(sw),26(sw),31(sw), 34	3,15(sw),33(sw),34
Case 3	Node 9 – Node 35	9, 6(sw),26(sw),35	9,11(sw),29(sw),35
Case 4	Node 2 – Node 30	2,1(sw),21(sw), 26(sw),30	2,8(sw),26(sw),29(sw), 30
Case 5	Node 12 – Node 34	12,11(sw),31(sw) ,34	12,18(sw), 36(sw), 34

Tables 3a and 3b show the node-node communication in case of 36 nodes topologies for all the 4 architectures

Table 4a:Communication paths for Raw and TriBA in case of 49 nodes topology

Communication paths for 49 Nodes			
	Source-Destination	Raw	TriBA
Case 1	Node 2 – Node 48	2,9,16,23,30,37,44,45,46,47,48	2,4,5,12,25,26,48
Case 2	Node 3- Node 47	3,10,17,24,31,38,45,46,47	3,2,4,5,10,12,25,26,46,47
Case 3	Node 10 – Node 48	10,17,24,31,38,45,46,47,48	10,12,25,26,26,46,48
Case 4	Node 2 – Node 42	2,3,4,5,6,7,14,21,28,35,42	2,4,5,10,11,22,23,40,42
Case 5	Node 13 – Node 47	13,12,19,26,33,40,47	13,14,12,25,26,46,47

Table 4b: Communication paths for LBDR and proposed architectures in case of 49 nodes topology

Communication paths for 49 Nodes			
	Source-Destination	LBDR	Proposed
Case 1	Node 2 – Node 48	2,1(sw),6(sw),11(sw),41(sw),46(sw),48	2,22(sw),43(sw),46(sw),48
Case 2	Node 3- Node 47	3,1(sw),6(sw),11(sw),16(sw),41(sw),46(sw),47	3,4(sw),7(sw),28(sw),49(sw),47
Case 3	Node 10 – Node 48	10,6(sw),11(sw),16(sw),41(sw),46(sw),48	10,17(sw),38(sw),41(sw),48
Case 4	Node 2 – Node 42	2,1(sw),6(sw),11(sw),16(sw),41(sw),42	2,4(sw),7(sw),28(sw),49(sw),42
Case 5	Node 13 – Node 47	13,11(sw),31(sw),36(sw),46(sw),47	13,12(sw),33(sw),47

Tables 4a and 4b shows the node-node communication in case of 49 nodes topologies for all the 4 architectures

Table 5a: Communication paths for Raw and TriBA in case of 64 nodes topology

Communication paths for 64 Nodes			
	Source-Destination	Raw	TriBA
Case 1	Node 2–Node 63	2,10,18,26,34,42,50,58,59,60, 61,62,63	2,4,6,15,31,33,61, 63
Case 2	Node 3-Node 56	3,4,5,6,7,8,16,24, 32,40,48,56	3,2,4,6,13,14,28,30, 55,56
Case 3	Node 12–Node 63	12,13,14,15,23, 31,39,47,55,63	12,14,15,31,33,61,63
Case 4	Node 2–Node 56	2,10,18,26,34,42,50,51,52,53, 54,55,56	2,4,6,13,14,28,30,55, 56
Case 5	Node 15 –Node 62	15,23,31,39,47,55,63,62	15,31,33,61,62

Table 5b: Communication paths for LBDR and proposed architectures in case of 64 nodes topology

Communication paths for 64 Nodes			
	Source-Destination	LBDR	Proposed
Case 1	Node 2 – Node 48	2,1(sw),6(sw),11(sw),16(sw), 41(sw),51(sw),61(sw),63	2,4(sw),7(sw),31(sw), 55(sw), 63
Case 2	Node 3- Node 47	3,1(sw),6(sw),11(sw),16(sw), 41(sw),51(sw),56	3,4(sw),28(sw),52(sw), 55(sw),56
Case 3	Node 10 – Node 48	12,11(sw),16(sw),41(sw),51(sw), 61(sw),63	12,10(sw),34(sw),58(sw), 61(sw),63
Case 4	Node 2 – Node 42	2,1(sw),6(sw),11(sw),16(sw), 41(sw),51(sw),56	2,4(sw),7(sw),31(sw), 55(sw),56
Case 5	Node 13 – Node 47	15,11(sw),16(sw),41(sw), 51(sw), 61(sw),62	15,16(sw),40(sw), 64(sw),62

Tables 5a and 5b show the node-node communication in case of 64 nodes topologies for all the 4 architectures.

All the above tables are used to evaluate and compare the performance of Proposed architecture with the existing architectures.

4.3 Output Parameters

As mentioned in the assumption section, while evaluating power consumption, the power consumed by the switch is considered to be less when compared with power consumed by the core node. The logical reason for this kind of assumption is the basic behavior of a computing component and switching component. When a packet arrives a switching component it will just check the source and destination fields in a packet header and some other small parameters like checksum to check if it is a valid packet or not. While a core checks the actual message and processes the whole detail of the packet. It will certainly consume more energy until the particular task is completed. While the switching component just checks the header and it will not bother about the details of the data that is transferred between source and destination nodes. Hence, if power consumption by a switching component is considered to be 1 unit, then the power consumption by a computing component is certainly greater than 1 (>1). Hence, it is taken as 2 units for computing component which is the next highest integer. The following sections show the evaluation parts using the above output parameters.

4.4 Comparison of Number of Switches

In this section of results the number of switches required to have full connectivity among cores are compared for all the analyzed designs in multi-core architecture. Proposed design is compared with Raw architecture, TriBA and design used to implement LBDR. From the graph in Figure 4.2, it can be analyzed that the proposed design requires less number of switches when compared with Raw and TriBA. But, when compared with the design that is used to implement LBDR, it requires more number of switches. Inspite of this drawback it can be seen in the next sections that the power consumption and communication delay is efficient in the proposed design than the latter.

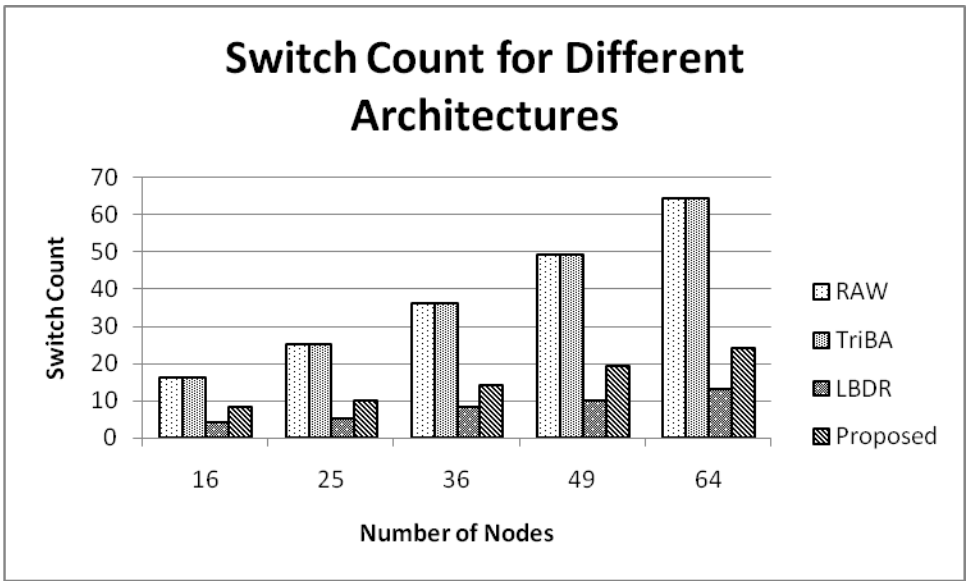

Figure 4.2: Graph Comparing the Number of Switching Components

Also, the main advantage of the proposed methodology over the other architectures that is used for LBDR is all the 4 cores are connected to only a single switch and if the switch goes bad all the cores will lost connectivity with the remaining cores. In case of proposed design it is taken care that each core is connected to minimum of 2 switches which provides alternate routes when one switch goes bad. And in most of the cases the computing components are connected to more than 2 switches.

4. 5 Comparison of Power Consumption

As per [30] there are 2 key requirements for the designers of multi-core architecture. The first requirement is network power, which is the amount of power consumed by the network nodes while they are up and running. Second requirement that the designers should consider while designing the multi-core architecture is Latency. Details about latency are described well in the next section.

Below is the graphical representation of the calculated results for power consumption in each case.

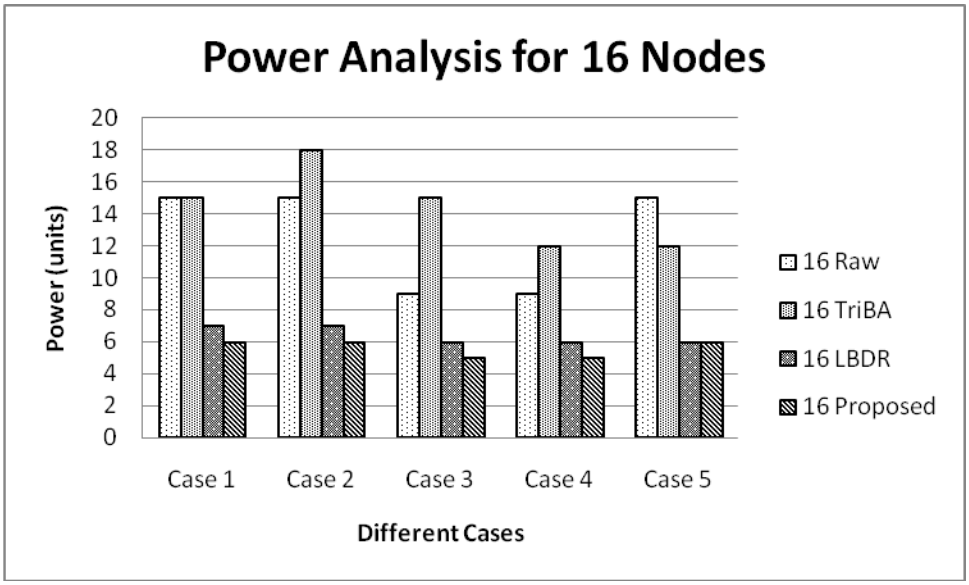

Figure 4.3:Power Analysis for 16 Nodes

The graph in Figure4.3, indicates that the power consumption in case of proposed architecture is less when compared to other 3 architectures. When calculated the bars in the graph indicates the exact number of units of power consumed for each design for 16 nodes.

Similarly power consumption is calculated for 25 nodes,36 nodes, 49 nodes and 64 nodes.

Following graphs shows the comparison and values for each combination.

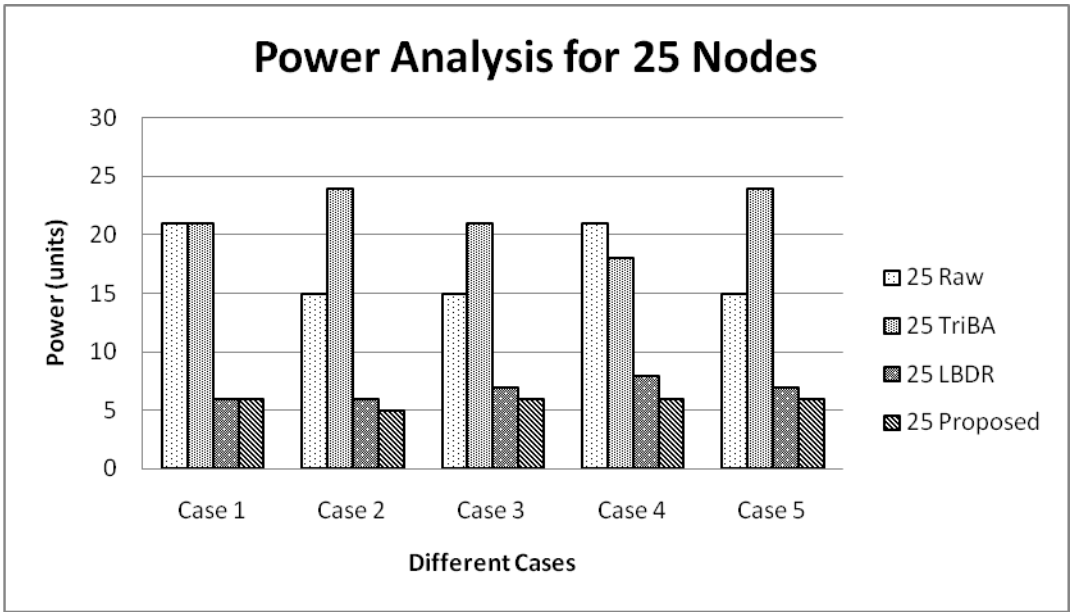

Figure 4.4:Power Analysis for 25 Nodes

The graph in Figure4.4, indicates the power consumption analysis for the selected 3 architectures and the proposed architecture in case of 25 nodes topology in each architecture.

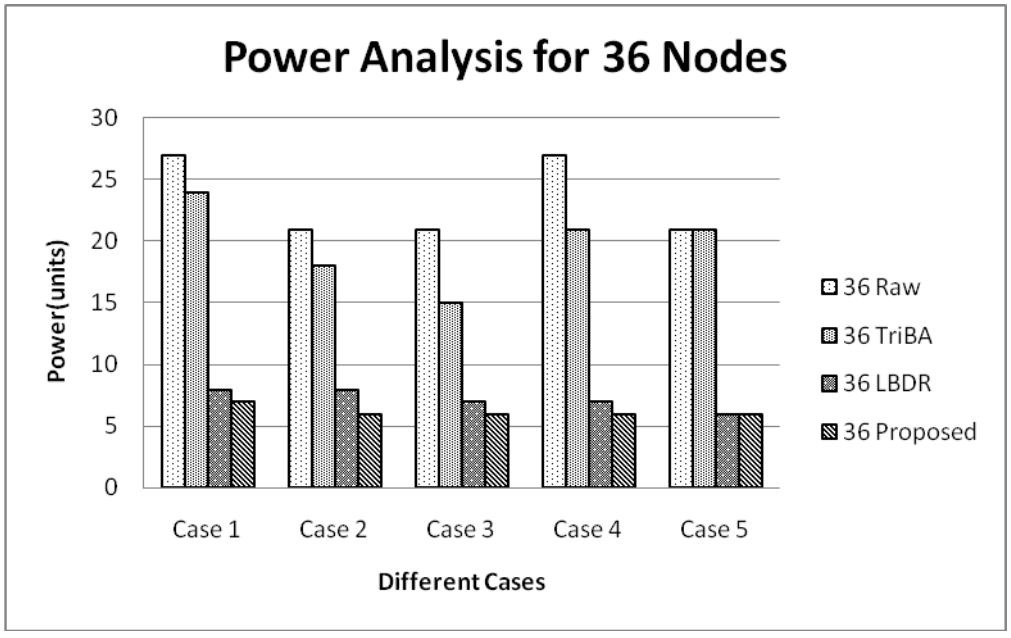

Figure 4.5:Power Analysis for 36 Nodes

The graph in Figure4.5, indicates the power consumption analysis for the selected 3 architectures and the proposed architecture in case of 36 nodes topology in each architecture.

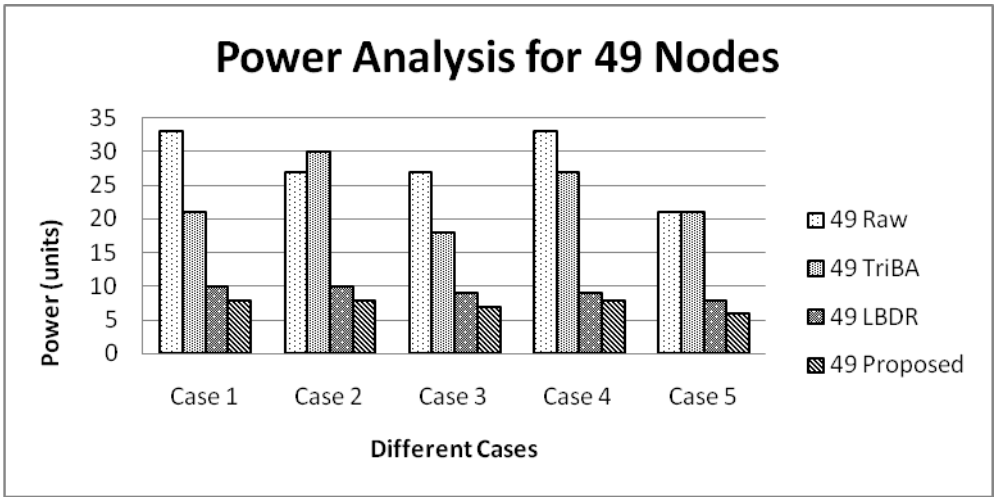

Figure 4.6:Power Analysis for 49 Nodes

The graph in Figure4.6, indicates the power consumption analysis for the selected 3 architectures and the proposed architecture in case of 49 nodes topology in each architecture.

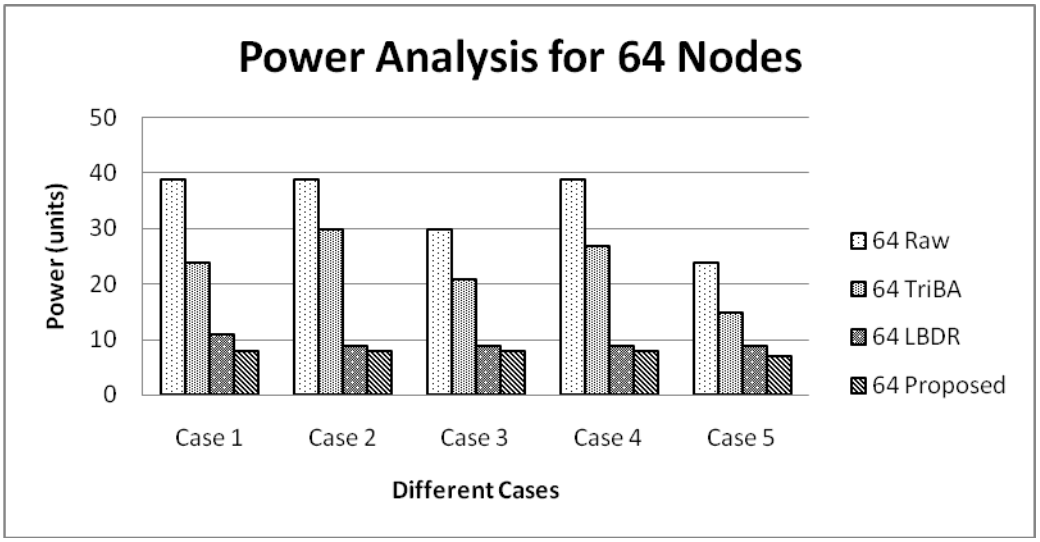

Figure 4.7: Power Analysis for 64 Nodes

The graph in Figure4.7, indicates the power consumption analysis for the selected 3 architectures and the proposed architecture in case of 25 nodes topology in each architecture.

Thus after going through all the graphs it can be observed that the power consumption in all the cases for all kinds of topologies is more efficient for proposed design when compared with remaining 3 architectures.

4.6 Comparison of Communication Delay

As described in the previous results section, in this section of results communication delay is calculated for different cases when cores at random places in a given topology are communicating. As discussed in the introduction section latency is a term which refers to the delay for a message to reach its destination while it traverses the path between the source and destination. Hence, it can be inferred that the less number of hops a message traverses from source to destination the less would be delay. Therefore in the section of results communication delay is measured in terms of number of hops required for a core to communicate with the other core in each of the analyzed cases. 5 cases are considered for 16, 36, 64 node networks. In each case all the values of the obtained for the proposed design are compared with Raw, TriBA and with the design used for LBDR. Values obtained are the number of hops. Following graphs show the comparison of the same. The cases considered for communication delay are same as the cases that are considered for power consumption.

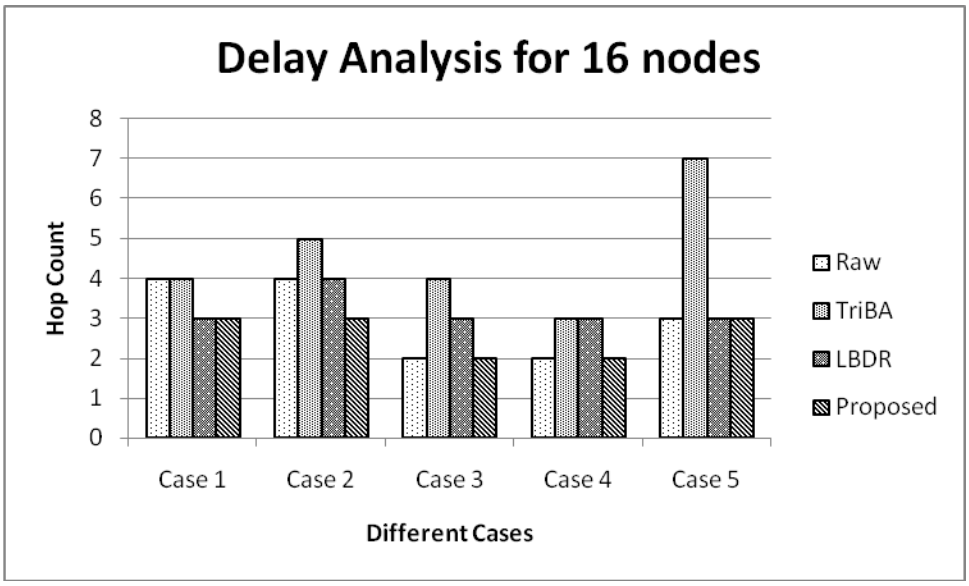

Figure 4.8: Delay Analysis for 16 Nodes

Figure 4.8, shows the number of hop counts required for a message between 2 different cores on the same chip in case of 3 selected architectures and the proposed architecture. As discussed before hop count determines the delay for a message to traverse from a source core to a destination core. The graph in Figure4.8, indicates the delay analysis for 16 nodes topology of all the 4 architectures.

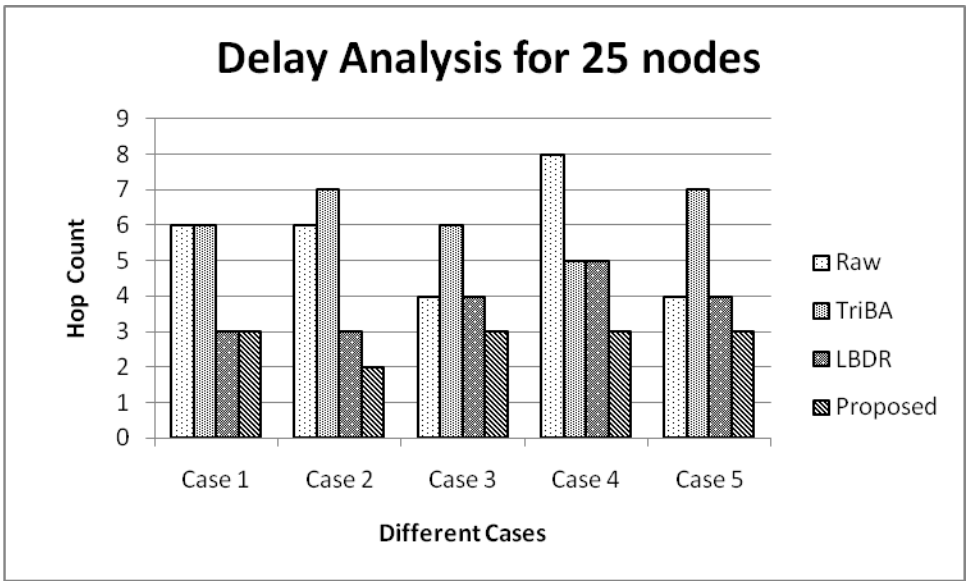

Figure 4.9:Delay Analysis for 25 Nodes

The graph in Figure4.9, indicates the delay analysis for 25 nodes topology of all the 4 architectures.

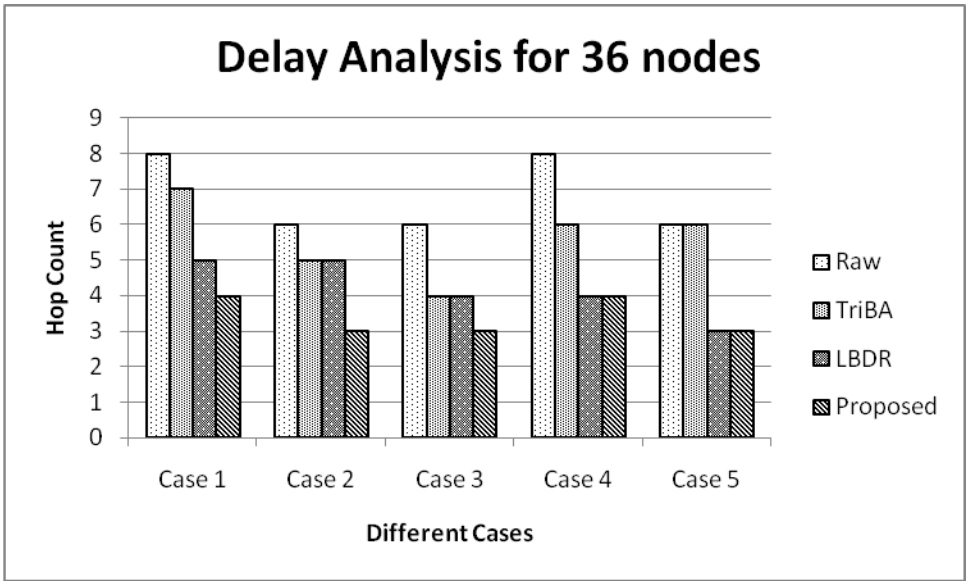

Figure 4.10: Delay Analysis for 36 Nodes

The graph in Figure4.10, indicates the delay analysis for 36 nodes topology of all the 4 architectures.

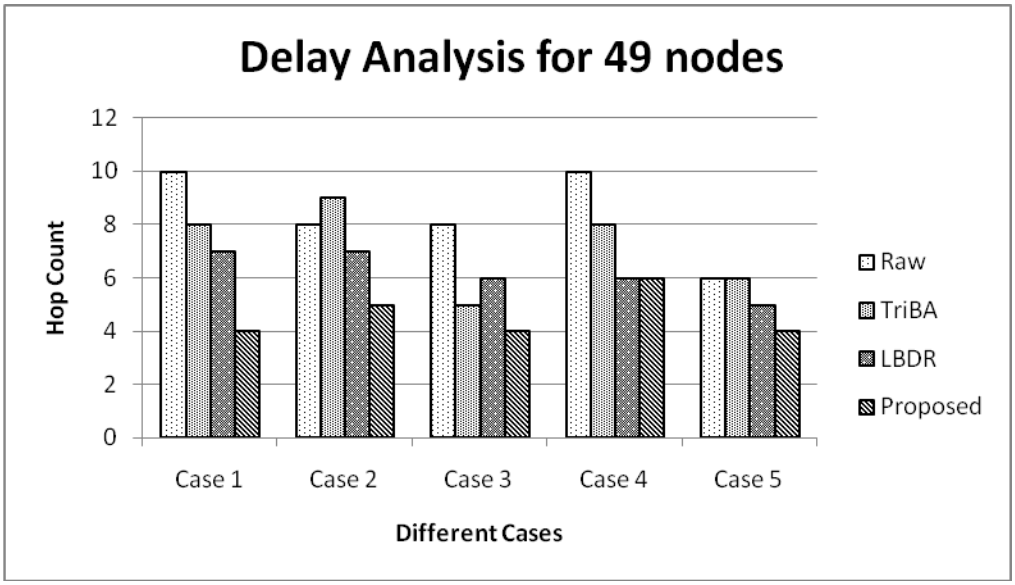

Figure 4.11: Delay Analysis for 49 Nodes

The graph in Figure4.11, indicates the delay analysis for 49 nodes topology of all the 4 architectures.

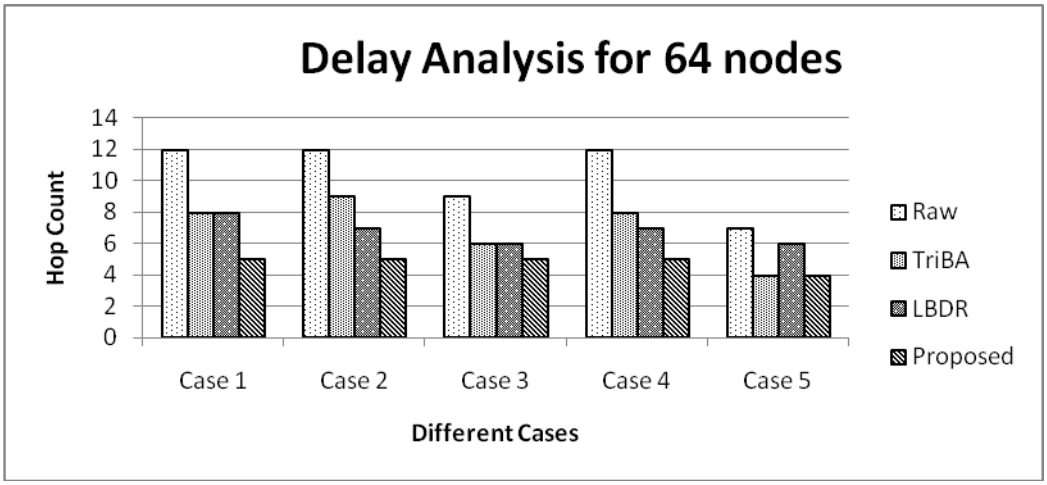

Figure 4.12:Delay Analysis for 64 Nodes

The graph in Figure4.12, indicates the delay analysis for 64 nodes topology of all the 4 architectures.

From the above results it can be concluded that the communication delay in case of the proposed design is less when compared with remaining 3 designs in multi-core architecture.

4.7 Summary and Observations

From the above evaluations we can summarize that proposed architecture performs better than the other 3 selected architectures. After analyzing all the available values that are used for evaluation it is observed that the power consumption in case of proposed architecture is approximately 77% lesser than the Raw architecture from MIT in case of 64 nodes mesh topology. Similarly, it is observed that the communication delay in case of proposed architecture is 54% lesser than the communication delay for Raw architecture from MIT. The comparison in terms of percentage is calculated by summation of all the available values in case of each architecture and by calculating the difference ratio. Hence, it can be analyzed that proposed architecture performs better than the other similar architectures without compromising the computational efficiency.

The following table summarizes the impact of the proposed architecture on number of switches, power consumption and communication delay when compared with the selected architectures. The percentage is calculated by considering the summation of the values that are obtained only for 64 nodes topologies for all the architectures.

Table 6: Comparison of Proposed Architecture with Raw, TriBA, and LBDR

	RAW	TriBA	LBDR
Number of Switches	(+) 62.5	(+) 62.5	(-) 50
Power Consumption	(+)77	(+) 67	(+)17
Communication Delay	(+) 54	(+)31	(+) 29

CHAPTER 5

CONCLUSION AND FUTURE WORK

We hope the discussion presented in the thesis motivates the interested scholars into considering research in the challenging but prosperous area of multi-core systems. Multi-core architecture is the future of all modern computing areas from server to desktop to embedded environments. With the appropriate architecture, the potential of multi-core systems can be enormous. Our contributions lead to solutions that overcome the disadvantages due to current poor core-to-core communication and the presence of caches in multi-core. In this chapter, we conclude our work and offer a list of possible future extensions of this work.

5.1 Conclusion

It is proven that multi-core architecture provides better performance/power ratio suitable for real-time applications. However, current multi-core system is not suitable to decrease power consumption and increase memory-level parallelism due to the wasteful core-to-core interconnection topology. For example, each node/core in MIT Raw architecture has computing and switching components. Computing component of such a node consumes power while the node is working (only) as a switching component and vice versa. Moreover, due to the presence of multiple level-1 caches (each core has its own private cache) multi-core architecture suffers from data inconsistency, power consumption, and heat dissipation.

In this paper, we propose a multi-core design methodology to reduce the number of switches without any negative impact on the performance. According to this method, nodes are separated

between computing cores and network switches. However, there are some special nodes (computing/switching nodes) with dual functionalities. Using folded torus concept, we develop an algorithm to determine the computing cores and network switches and how to connect them (cores and switches) in the multi-core architecture. Multi-core architectures with various numbers of nodes (cores and switches) are used to evaluate the proposed methodology. We obtain the core-to-core communication delay and total power consumption for MIT Raw, Triplet Based Architecture (TriBA), Logic-Based Distributed Routing (LBDR), and the proposed architecture using synthetic workload. In addition, we collaborate with other students to develop a simulation platform for multi-core systems.

According to the experimental results, the proposed architecture outperforms Raw, TriBA, and LBDR by cutting down the number of switches significantly. Average delay is decreased due to the fact that each switch provides adequate communication channels. Total power consumption is reduced as the number of switches is cut down. Based on the results, proposed architecture may reduce the total power consumption by up to 77% and average delay by up to 54%. It is also noted that the communication is more reliable in the proposed architecture because each computing core is connected to multiple switches.

5.2 Future Extensions

Our thesis contributions including the design methodology to reduce the number of switches in multi-core architectures can be extended to cope with the following important research areas.

- Efficient routing algorithms for multi-core systems: Develop routing tables for the switches and propose efficient routing algorithms for multi-core systems for reliable communication with minimal delay.

- Multi-core modeling and simulation platform support: Modeling and simulation platforms are important to analyze multi-core systems. Proposed methodology can be extended to assist developing and/or evaluating multi-core modeling and simulation platforms.

- Evaluate core allocation strategies in multi-core: Effective core allocation in multi-core architecture may significantly reduce heat intensity of a multi-core chip. Proposed methodology can be extended to measure the impact of various core allocation strategies on power consumption and heat dissipation of multi-core architecture.

REFERENCES

LIST OF REFERENCES

[1] Chatti, Majed; Yehia, Sami; Timsit, Claude; Zertal, Soraya; 2010 International Conference on High Performance Computing and Simulation (HPCS), page(s): 623–630. DOI: 10.1109/HPCS.2010.5547065.2010.

[2] Jin Liu; Delgado-Frias, J.G.; Xiaofeng Wang; "A Novel Analytical Model for Wormhole Switching Network on Chip with Adaptive Routing" , 2010 53rd IEEE International Midwest Symposium on Circuits and Systems (MWSCAS), page(s): 733–736. DOI: 10.1109/MWSCAS.2010.5548715. 2010.

[3] Freitas, H.C.; Santos, T.G.S.; Navaux, P.O.A.; "Design of programmable NoC router architecture on FPGA for multi-cluster NoC" Electronics Letters Volume: 44, Issue: 16,page(s): 969–971. DOI: 10.1049/el:20080854. 2008.

[4] Li-ShiuanPeh, Stephen W. Keckler, and SriramVangal; "On-Chip Networks for Multicore Systems"; Springer Science+Business Media, LLC 2009,page(s) 35-71. doi: 10.1007/978-1-4419-0263-4. 2009.

[5] D.K. Every. "IBM's Cell Processor: The next generation of computing".Shareware Press, 2005, http://www.mymac.com/fileupload/CellProcessor.pdf (accessed in October 2011).

[6] S. Rader, J. Corleto-Mena, N. Marshall, et al. "Mobile Extreme Convergence: A Streamlined Architecture to Deliver Mass-Market Converged Mobile Devices"; Freescale Semiconductor.2005.

[7] P. Ranganathan, S. Adve, and N.P. Jouppi. "Reconfigurable Caches and their Application to Media Processing." ISCA/ ACM, page(s) 214–224, Vancouver, Canada. 2000.

[8] P. Reed, M. Alexander, et al (Motorola). "A 66-MHz Configurable Secondary Cache Controller with Primary Cache Copy-back Support". IEEE-1992, page(s) 16-17. 1992.

[9] V. Romanchenko. "Evaluation of the multicore processor architecture Intel core: Conroe", Kentsfield, Digital-Daily.com. 2006.

[10] T. Tian. Intel Corp. "Effective Use of the Shared Cache in Multicore Architectures". Dr. Dobb's Portal, 2007.

[11] Q. Xu and P.J. Teller. Unified vs. split TLBs and caches in shared-memory MP systems. 9th International Parallel Processing Symposium page(s) 398. 1995

[12] Manira S. Rani "An Efficient and scalable core allocation strategy for Multi-core systems", Thesis in Masters of Science, Florida Atlantic University, May, 2011.

[13] Jing-Mei Li; Ping Jiao; Chao-GuangMen; "The Heterogeneous architecture of Multi-Core research and design" , MASS '09. International Conference on Management and Service Science , 2009.doi: 10.1109/ICMSS.2009.5302477 Publication Year: 2009, page(s): 1 – 6. 2009.

[14] McNairy C, Bhatia R. Montecito: "A Dual-core, Dual-thread Itanium Processor" [J]. IEEE Micro, 25(2): 10-20.2005.

[15] Intel® Multi-Core Processor. 2011. www.intel.com/software/enterprise, (accessed in September, 2011).

[16] Intel_ Core™ Microarchitecture. 2011. www.intel.com/Multi-Core, (accessed in September, 2011).

[17] Bryan O'Sullivan, Don Stewart, and John Goerzen; "Real World Haskell". 2011. http://book.realworldhaskell.org/read/concurrent-and-multicore-programming.html.

[18] "An Effective Approach for Multicast on Multi-core Architecture "Yuxin Wang; Liye Yuan; He Guo; XinzhongHui; Yuansheng Yang; Scalable Computing and Communications; Eighth International Conference on Embedded Computing, 2009. SCALCOM-EMBEDDEDCOM'09, page(s): 37 – 41. doi: 10.1109/EmbeddedCom-ScalCom.2009.17.2009.

[19] Phi-Hung Pham; Phuong Mau; Chulwoo Kim; "A 64-PE Folded-Torus Intra-chip Communication Fabric for Guaranteed Throughput in Network-on-Chip Based Applications " , Custom Integrated Circuits Conference, 2009. CICC '09. IEEE doi:10.1109/CICC.2009.5280748, page(s): 645 – 648. 2009.

[20] K. C. Chang, J. S. Shen and T. F. Chen, "Evaluation and Design Trade-offs between Circuit-Switched and Packet-Switched NOCs for Application-Specific SOCs", Design Automation Conference, pp. 143-148,July 2006.

[21] Prototype Design of Hybrid Multi-Core Architecture for Real-Time Application Computer Engineering and Technology (ICCET), 2010 2nd International Conference on ,Vol: 1 doi: 10.1109/ICCET.2010.5486080, page(s): V1-404 - V1-408. 2010.

[22] Yaghini, P.M.; Eghbal, A.; Pedram, H.; Zarandi, H.R.; Parallel, "Asynchronous NOC Router Design" Distributed and Network-Based Processing (PDP), 2010 18th Euromicro International Conference on, doi: 10.1109/PDP.2010.21, page(s): 540 – 545. 2010.

[23] Rodrigo, S.; Medardoni, S.; Flich, J.; Bertozzi, D.; Duato, J.; "Efficient implementation of distributed routing algorithms for NoCs Computers & Digital Techniques, IET Volume: 3, Issue: 5, doi: 10.1049/iet-cdt.2008.0092, page(s): 460-475.2009.

[24] Raw Architecture Workstation. 2011. http://groups.csail.mit.edu/cag/raw/purpose, (accessed in April, 2011).

[25] Michael B. Taylor, Walter Lee, et al.; "Tiled Multicore Processors"; Springer Science+Business Media, LLC.doi: 10.1007/978-1-4419-0263-4, pp.1-34.2009.

[26] Michael Bedford TaylorA.B., "Design Decisions in the Implementation of a Raw Architecture Workstation", Dartmouth College 1996.Masters in science, Massachusets Institute Of Technology. 2011.

[27] Haroon-Ur-Rashid; Shi Feng; JiWeixing; "Triplet Based Multi-core Interconnection Network and its Computational Efficiency", Computer and Information Science, 2009. ICIS 2009. Eighth IEEE/ACIS International Conference on, doi: 10.1109/ICIS.2009.137, page(s): 516 – 521.2009.

[28] Abu Asaduzzaman, et al.; "On the Design of Low-Power Cache Memories for Homogeneous Multi-Core Processors"; IEEE 22nd International Conference on Microelectronics (ICM'10), page(s) 387-390.2010.

[29] Abu Asaduzzaman, et al.; "Modeling Multicore Distributed Systems and Simulation of Performance, Power, and Predictability using VisualSim"; Huntsville Simulation Conference (HSC-2008) sponsored by SCS and hosted by AMSC, Huntsville, Alabama, USA. 2008.

[30] Stephen W. Keckler, KunleOlukotun, and H. Peter Hofstee; "Multicore Processors and Systems"; Springer Science+Business Media, LLC. 2009. DOI:10.1007/978-1-4419-0263-4